Q33：搜索旋转排序数组：一个没有重复元素的有序数组， 按照某一点发生了旋转， [0,1,2,4,5,6,7] -》[4,5,6,7,0,1,2]

1. 采用二分法： 若左侧有序，则若target在左侧就搜索左侧，否则搜右侧。 右侧同理。

Q34 . 在排序数组中查找元素的第一个和最后一个位置 要求： o(logn)

1. a[mid] = target a[mid] < target a[mid] > target
2. 分为三种情况进行二分 a[mid] = target 的写法要先看左右两侧还有没有target.

Q50：Pow(x, n)：二分快速幂

很简单了，没啥可讲的了。

Q74：搜索二维矩阵：00 到n,m有序。 搜索 target。

1. 二分搜索最后一列， 搜索第一个大于target的值， 如果没找到，那么就是不存在。
2. 如果找到第0行，那么若大于第0行第0个元素，那么target就可能在第0行。否则就是小于最小的不存在。
3. 找到在哪一行，然后对该行进行二分搜索。找不到就是不存在。
4. 该题或则一个类似的题：可以从左下角搜索，target。 小了，就向右，大了，就向上。

Q75： 颜色分类：一个只包含3个颜色的数组，按照 红白蓝进行排列。

三分排列

1. 对于一个target值， 左侧都是小于target的， 右侧都是大于target的。该方法是可以的。而且很快。
2. 双指针。left指向当前应该放红色的位置，right指向当前应该放蓝色的地方。然后用i在left到right遍历。
3. 注意：i遇到0要和前边交换，遇到1直接划过。 但是遇到2要注意，和后边交换后，换回来的仍然可能是
   2. 因为i是从做到右滑动的只能保证i之前没有1，不能保证i之后没有2.

Q162： 寻找峰值(任意一个) 假设 位置-1和len的大小是负无穷 。最简单的做法就是 遍历一遍，第一个和最后一个特殊处理，即可。但是本题要求时间o(lgn)

```
while (l < r) {
    int mid = (l + r) / 2;
    if (nums[mid] > nums[mid + 1])
        r = mid;// mid位置就可能时山峰。在左侧
    else
        l = mid + 1; // mid+1的位置就可能是山峰，在右侧。
}
```

Q215 数组中的第K个最大元素

快速查找算法。写了很多了，就不说了。